A LOW OVERHEAD FAULT TOLERANT AND CONGESTION AWARE ROUTING ALGORITHM

1.PAPPULA JAYASRILAKSHMI, 2. TIKKIREDDI ADITYA KUMAR
1. M.Tech Student, Dept. of ECE, Prasiddha College of Engineering & Technology, Anathavaram,AP
2. Associate Professor, Dept. of ECE, Prasiddha College of Engineering & Technology, Anathavaram,AP.

ABSTRACT:
Network-On-Chip (NoC) has surpassed the traditional bus based on-chip communication in offering better performance for data transfers among many processing, peripheral and other cores of high performance embedded systems. Adaptive routing provides an effective way of efficient on-chip communication among NoC cores. The message routing efficiency can further improve the performance of NoC based embedded systems on a chip. Congestion awareness has been applied to adaptive routing for achieving better data throughput and latency. This thesis presents a novel approach of analyzing congestion to improve NoC throughput by improving packet allocation in NoC routers. The routers would have the knowledge of the traffic conditions around themselves by utilizing the congestion information. We employ header flits to store the congestion information that does not require any additional communication links between the routers. By prioritizing data packets that are likely to suffer the worst congestion would improve overall NoC data transfer latency. Further, this project is enhanced by using more secured algorithm like Scalable Encryption technique.

INTRODUCTION:
Embedded systems on chip in the recent decade have grown substantially utilizing many cores on a single chip known as embedded System-on-a-chip (SoC). It is important that data within the SoC, all cores have access to the desire resources while maintaining a data load balanced transmission links. Traditional bus-based interconnection has been able to allow high speed data transfers within a small SoC. However as the SoC scales up in size in today‟s most demanding applications, NoC is proven to provide better balance between traffic loads and data access for every core. [1] NoCs are expandable to allow communication for very large SoCs. It is not uncommon to see more than 256 cores in a NoC system, which is impossible to handle on a bus-based system. NoC based systems is a new strategy for data communication within the SoC. NoC performance depends on topology, data link width, traffic patterns, routing mechanisms and router arbitration. These parameters can be prioritized to improve performance, area of the design layout or power dissipation of the SoC. Depending on the chosen design requirement, the connections between routers and cores of a NoC can differ significantly. 2 NoC performance is important in high data throughput applications. It is ideal to reduce congestion and latency for all the data transfers within the NoC. Unfortunately, by increasing the number of cores within the NoC, congestion increases proportionally to the NoC size. Therefore, it is necessary to manage traffic effectively for reducing congestion within the NoC to achieve the best performance as the NoC grow in size. Design and scalability issues associated with increasing core counts on Chip Multi-Processors (CMPs) is a prominent research domain in computer architecture over the last decade. Communication among cores in these CMPs housing processors, caches and memory controllers is an important task that requires deeper exploration for better performance and throughput. Thus designing a scalable interconnect is critical for future energy efficient CMP designs. Interconnects like bus and crossbars are no longer scalable with these ever growing trend in CMPs. Hence, researchers have moved towards Network-onChip (NoC), a scalable, packet switched and distributed interconnect framework that offer much lower latency and higher bandwidth than their traditional bus based counter parts. Most modern CMPs are arranged in 2D mesh topology due to its simple layout and short wires. Our approach here is to design a variable hardware router code by using Verilog and the same to be implemented for the SOC (System On Chip) level router.

CONGESTION AWARENESS: Congestion awareness can be catorgorized in three types, locally adaptive, regionally adaptive and globally adaptive. The amount of congestion data received by each router depends on the type of congestion awareness chosen. Locally adaptive does not need traffic flow between routers as the amount of credits available from the credit value registers (CVRs) is sufficient while globally adaptive need a large amount of data transfer to hold all the congestion values to each downstream router. Congestion awareness generally sacrifices data transfer to determine a better path for a packet to route to its destination.
The OCP readily adapts to support new core capabilities while limiting test suite modifications for core upgrades.

Basically OCP has the address is of 13bits, data is of 8bits, control signal is of 3bits and burst is of integer type. The 8kbit memory (2^{13} = 8192bits = 8kbits) is used in the slav e side in order to verify the protocol functionality. The System will give the inputs to OCP Master during Write operation and receive signals from OCP Slave during Read operation.

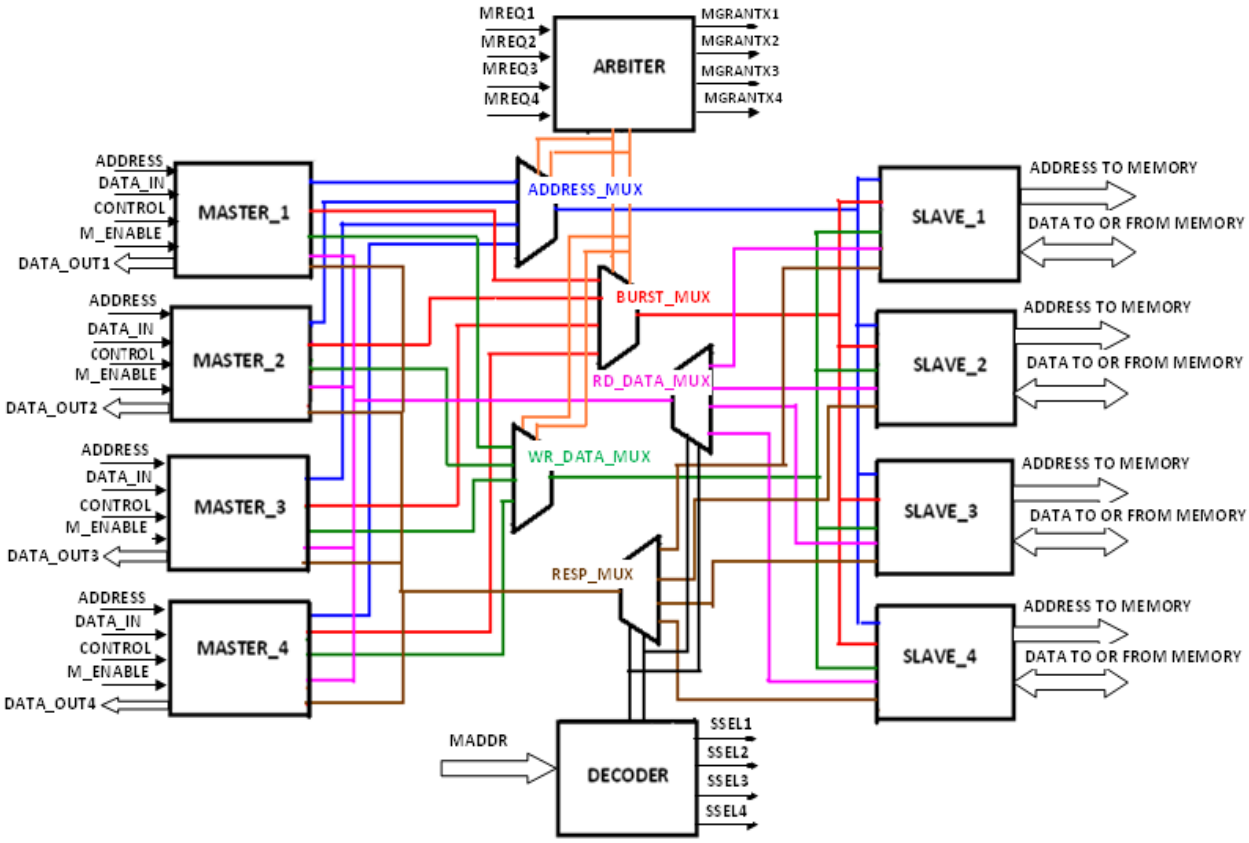

FIGURE : BLOCK DIAGRAM

ON-CHIP BUS FUNCTIONALITIES

The On-Chip Bus Functionalities are classified into 4 types including 1) burst, 2) lock, 3) pipelined, and 4) out-of-order transactions.

Burst transactions

The burst transactions allow the grouping of multiple transactions that have a certain address relationship, and can be classified into multi-request burst and single-request burst according to how many times the addresses are issued. FIGURE 3.2 shows the burst read transactions. The multi-request burst as defined in AHB where the address information must be issued for each command of a burst transaction (e.g., A11, A12, A13 and A14). This may cause some unnecessary overhead. In the more advanced bus architecture, the single-request burst transaction is supported. The burst type defined in AXI, the address information is issued only once for each burst transaction. In our proposed bus design we support both burst transactions such that IP cores with various burst types can use the proposed on-chip bus without changing their original burst behavior.

FIGURE 3BURST TRANSACTIONS

Lock transactions

Lock is a protection mechanism for masters that have low bus priorities. Without this mechanism the read/write transactions of masters with lower priority would be interrupted whenever a higher-priority master issues a request. Lock transactions prevent an arbiter from performing arbitration and assure that the low priority masters can complete its granted transaction without being interrupted.

Pipelined transactions (outstanding transactions)

For example, D21 is sent right after A21 is issued plus *t*. For a pipelined transaction as shown in below FIGURE, this hard link is not required. Thus A21 can be issued right after A11 is issued without waiting for the return of data requested by A11 (i.e., D11-D14).

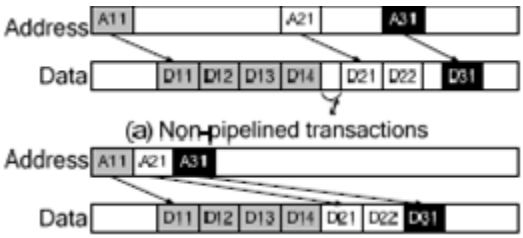

FIGURE PIPELINED TRANSACTIONS

Out-of-order transactions

The out-of-order transactions allow the return order of responses to be different from the order of their requests. These transactions can significantly improve the communication efficiency of an SOC system containing IP cores with various access latencies as illustrated in FIGURE 3.4 In FIGURE 3.4(a) which does not allow out-of-order transactions, the corresponding responses of A21 and A31 must be returned after the response of A11. With the support of out-of-order transactions as shown in FIGURE 3.4(b), the response with shorter access latency (D21, D22 and D31) can be returned before those with longer latency (D11-D14) and thus the transactions can be completed in much less cycles.

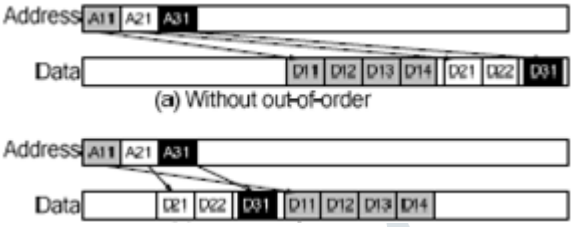

FIGURE OUT-OF-ORDER TRANSACTIONS

HARDWARE DESIGN OF ON-CHIP BUS

The architecture of the proposed on-chip bus is illustrated in FIGURE 3.5, where we show an example with two masters and two slaves. A crossbar architecture is employed such that more than one master can communicate with more than one slave simultaneously. If not all masters require the accessing paths to all slaves, partial crossbar architecture is also allowed. The main blocks of the proposed bus architecture are described next.

ARBITER

In traditional shared bus architecture, resource contention happens whenever more than one master requests the bus at the same time. For a crossbar or partial crossbar architecture, resource contention occurs when more than one master is to access the same slave simultaneously. In the proposed design each slave IP is associated with an arbiter that determines which master can access the slave.

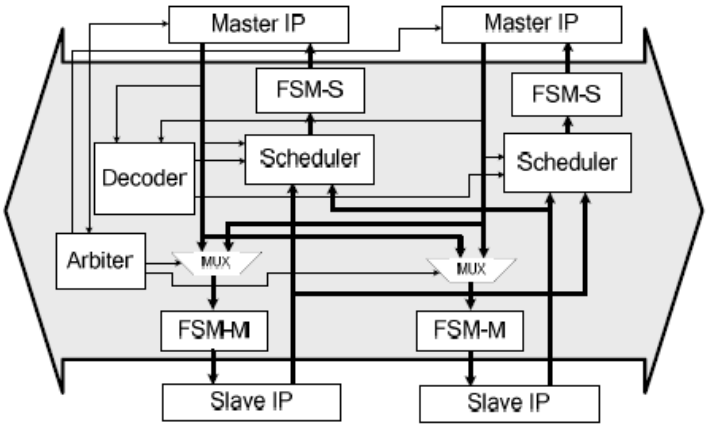

FIGURE TOPVIEW OF PROPOSED BUS ARCHITECTURE

Decoder

Since more than one slave exists in the system, the decoder decodes the address and decides which slave return response to the target master. In addition, the proposed decoder also checks whether the transaction address is illegal or nonexistent and responses with an error message if necessary.

FSM-M & FSM-S

Depending on whether a transaction is a read or a write operation, the request and response processes are different. For a write transaction, the data to be written is sent out together with the address of the target slave, and the transaction is complete when the target slave accepts the data and acknowledges the reception of the data. For a read operation, the address of the target slave is first sent out and the target slave will issue an accept signal when it receives the message. The slave then generates the required data and sends it to the bus where the data will be properly directed to the master requesting the data. The read transaction finally completes when the master accepts the response and issues an acknowledge signal. In the proposed bus architecture, we employ two types of finite state machines, namely FSM-M and FSM-S to control the flow of each transaction. FSM-M acts as a master and generates the OCP signals of a master, while FSM-S acts as a slave and generates those of a slave. These finite state machines are designed in a way that burst, pipelined, and out-or-order read/write transactions can all be properly controlled.

SCALABLE ENCRYPTION ALGORITHM:

SEAn,b operates on various text, key, and word sizes. It is based on a Feistel structure with a variable number of rounds, and is defined with respect to the following parameters:

• n plaintext size, key size;

• b processor (or word) size;

• nb = n/2b number of words per Feistel branch;

• nr number of block cipher rounds. As an only constraint, it is required that n is a multiple of 6b (Because both the plain text are separated into 2 parts, and all the operation are done in 3 words). Example- using 8-bit processor, we can derive a 48-bit block ciphers, denoted as SEA48, 8. Let x be a n/2-bit vector. We consider the following two representations.

• Bit representation: x b = x ((n/2)-1)…… x(2) x(1) x(0).

• Word representation: x w = x nb-1 x nb-2 …… x2 x1 x0.

Due to its simplicity constraints, SEAn,b is based on a limited number of elementary operations (selected for their availability in any processing device) denoted as follows: 1) Bit wise XOR 2) Mod 2b addition 3) A 3-bit substitution box S: = [0, 5, 6, 7, 4, 3, 1, 2] that can be applied bit wise to any set of 3-bit words for efficiency purposes. In addition, we use the following rotation operations: 4) Word rotation R, defined on nb-word vectors R: x y = R (x) yi+1 = x i 0<=I<=nb-2 y0 = x n b -1 5) Bit rotation r, defined on nb-word vectors R: x y = r (x) y = x3i >>>1 y 3i+1 = x 3i+1 y 3i+2 = x 3i+2 <<<= i <=(nb/3) –1 and –1 and >>> and << respectively.

LOOP ARCHITECTURE OF SEA:

The structure of our loop architecture for SEA is depicted in Fig, with the round function on the left part and the key schedule on the right part. Resource-consuming blocks are the S boxes and the mod2b adder; the Word Rotate and Bit Rotate blocks are implemented by swapping wires.

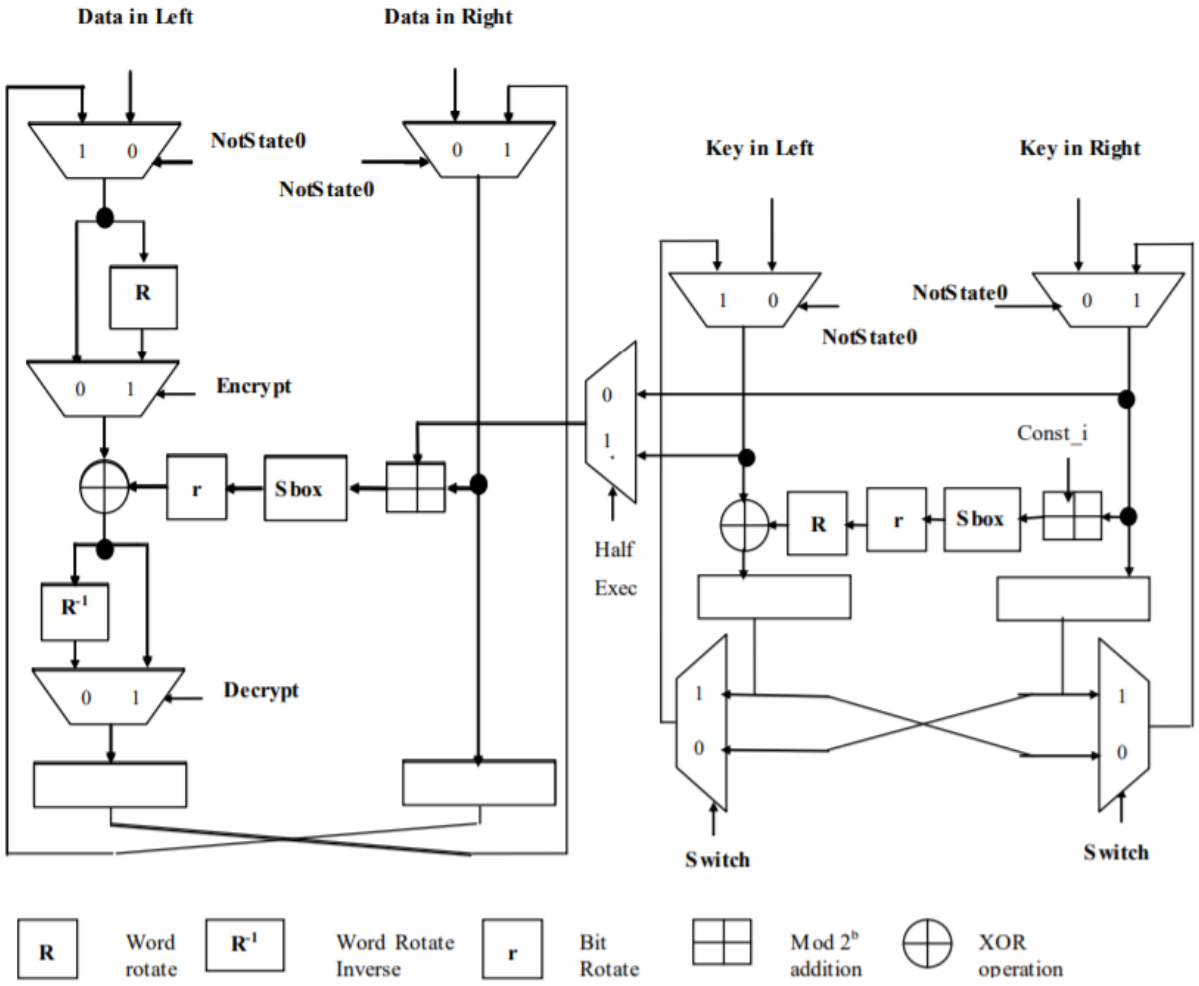

Fig . Loop architecture for SEA

According to the specifications, the key schedule contains two multiplexors allowing to switch the right and left part of the round key at half the execution of the algorithm using the appropriate command signal Switch. The multiplexor controlled by Half Exec provides the round function with the right part of the round key for the first half of the execution and transmits its left part instead after the switch. To support both encryption and decryption, finally added two multiplexors controlled by the Encrypt signal. Supplementary area consumption will be caused by the two routing paths. In the round function, the mod 2 adders are realized by using nb, b-bits adders working in parallel without carry propagation between them. In the key schedule, the signal Const_i (provided by the control part) can only take a value between 0 and nr/2.

RESULT:

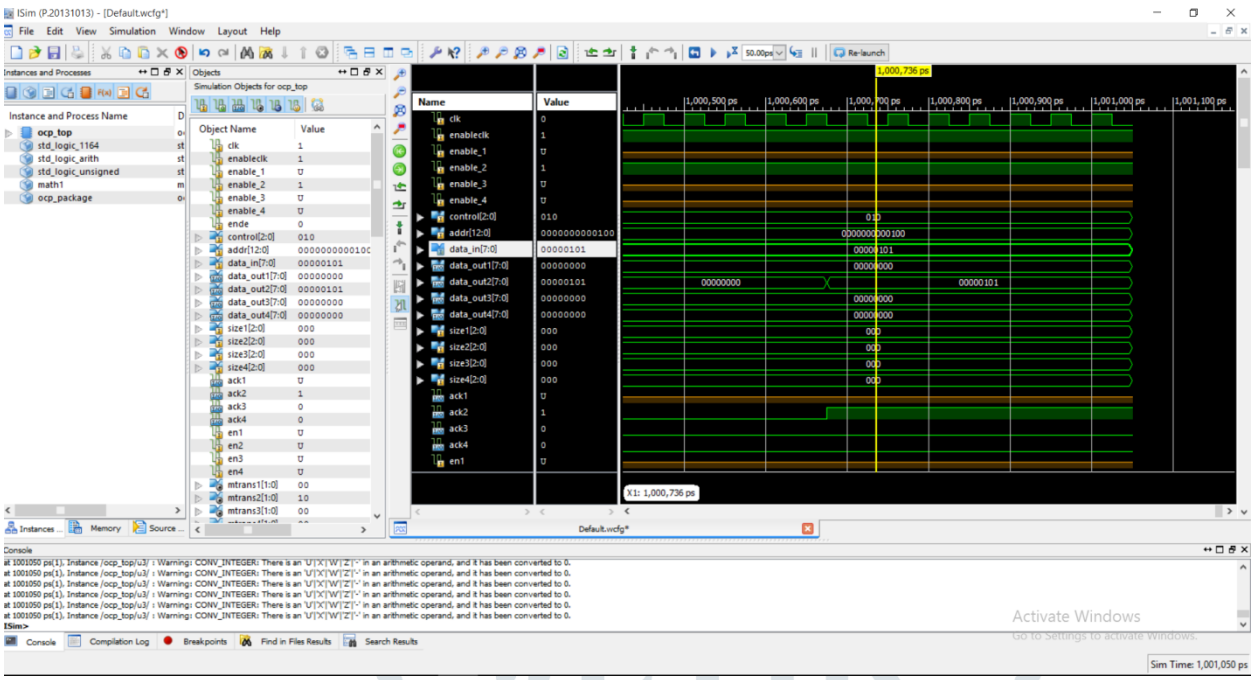

Fig: Existing technique

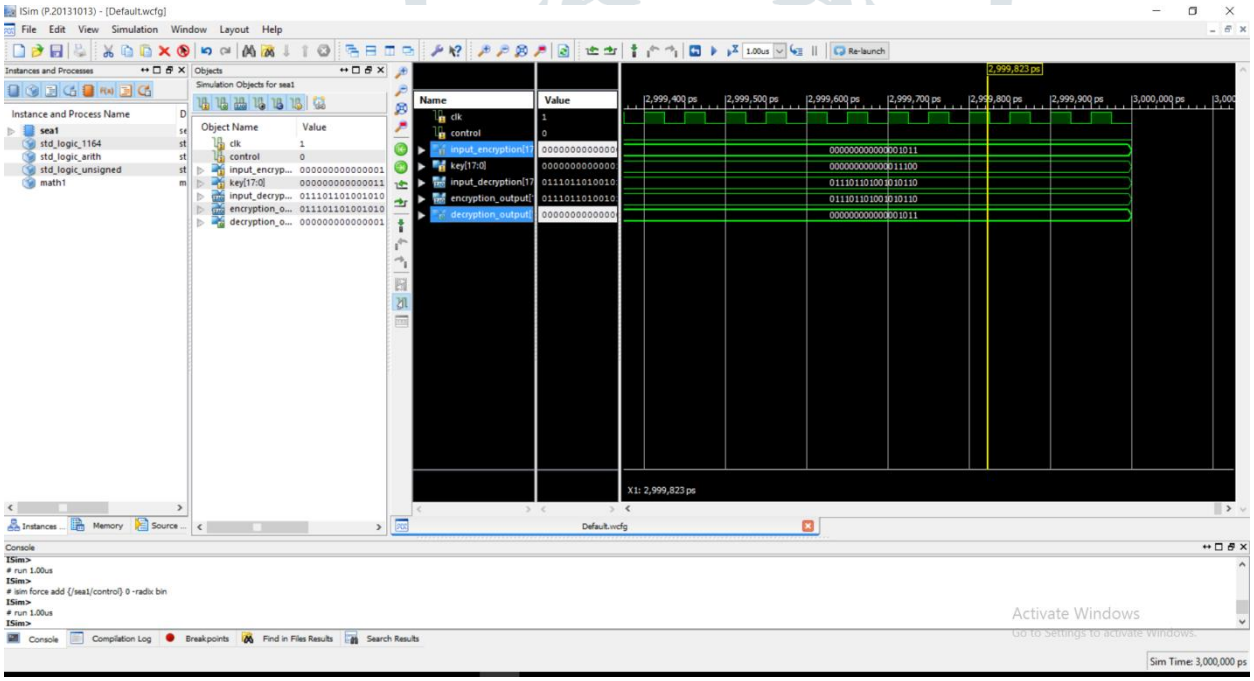

Fig: Proposed SEA algorithm

ADVANTAGES:

architecture includes the Test case which will define the test. The input driver block will generate the test input signals for the system testing. The input and output transacted will make the system get the input and output according to the system core requirement. The input and out monitor are placed to compare the system testing. At last the Response checker is to give the system testing pass or fail. The DUT is as shown below. General purpose router Router hardware code is variable More secured Robust router can handle all type of packets (Implemented on IPv4 and IPv6)

APPLICATION:

Can be used as public internetworking router Can be used as corporate router Software company private router (client to client, developer to client) Router for networking research In other words one point networking solution.

CONCLUSION:

This concept presents a novel approach that improves NoC throughput by packet prioritization. The objective is to increase the NoC throughput by using congestion aware information. Congestion awareness information has already been applied for on-chip communication to improve NoC routing. This project expanded adaptive routing by employing regional congestion data to latency for packet routing. Efficient enhancement described the methodology of expanding the regional congestion awareness data to improve packet selection for both VC and switch allocators. A new methodology of Congestion Aware Adaptive Routing (CAAR) is designed to prioritize the packet/flit allocation that suffers the most latency while travelling between NoC cores. Moreover, to improve on hardware usage and to avoid any additional links between routers, CAAR removes the sideband network to transfer congestion data and instead adds the congestion information into the header flit of a packet.

REFERENCES:

[1] W. Dally and B. Towles, Principles and Practices of Interconnection Networks. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003.

[2] L. Peh N. Enright Jerger, On-Chip Network, Synthesis Lectures on Computer Architecture.: Morgan & Claypool Publishers, 2009.

[3] G.-M. Chiu, "The Old-Even Turn Model for Adaptive Routing," in IEEE Transactions on Parallel and Distributed Systems, vol. 11, no. 7, pp. 729-738, July 2000.

[4] B. Grot, S.W. Keckler P. Gratz, "Regional Congestion Awareness for Load Balance in Network-on-Chip," High performance Computer Architecture (HPCA), pp. 203-214, 2008.

[5] B. Lin R.S. Ramanujam, "Destination-Based Adaptive Routing on 2D Mesh Networks," in Proceedings of Architectures for Networking and Communications Systems (ANCS), 2010 ACM/IEEE Symposium on, pp. 1-12, 25-26 October 2010.

[6] N. Enright Jerger, Z. Wang S. Ma, "DBAR: An Efficient Routing Algorithm to Support Multiple Concurrent Applications in Network-on-Chip," in Proceedings of Computer Architecture (ISCA), 2011 38th Annual International Symposium on, pp. 413-424, 4-8 June 2011.

[7] T. Takabatake, "Simulations of NoC topologies for generalized hierarchical completely-connected networks," in Proceedings of the 6th International Workshop on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), vol. , no. , pp. 1-5, 20-22 Jun. 2011.

99

[8] A., Saoud, S. Achballah, "A Survey of Network-On-Chip Tools," International Journal of Advanced Computer Science and Applications (IJACSA), vol. 4, no. 9, pp. 61-67, Dec. 2013.

[9] T., Mahadevan, S. Bjerregaard, "A Survey of Research and Practices of Network-on-Chip," ACM Computing Surveys, vol. 38, no. 1, pp. 1-51, Jun 2006.

[10] P. Bogdan, G. Wei, C.Y. Tsui, R. Marculescu Z. Qian, "A Traffic-aware Adaptive Routing Algorithm on a Highly Reconfigurable Network-on-Chip Architecture," in Proceedings of International Conference on Hardware/Software Codesign.